REMARKS

Claims 1-15, 27-32, 42 and 43 are pending, of which claims 1, 27 and 42 are independent. Favorable reconsideration and further examination are respectfully requested.

§ 112 Rejection

Claims 30, 31, and 43 have been amended. The Applicant believes that the amendments overcomes the Examiner's objections. Accordingly, withdrawal of the § 112 rejections is respectfully requested.

§ 101 Rejection

Claim 1, as amended, is directed to statutory subject matter. Claims 2-15 depend directly or indirectly from independent claim 1. Accordingly, withdrawal of the § 101 rejection is respectfully requested.

§ 102 Rejection

Independent claims 1, 27, and 42 were rejected under 35 USC § 102(b) as being anticipated by U.S. Patent No. 6,958,973 ("*Chen*"). Applicants respectfully traverse this rejection.

Claim 1 recites, in part, dynamically binding an *event context* to an *execution context* in response to receiving events." (emphasis added) A non-limiting example[1] of this feature of claim 1 is provided in the Applicant's specification as follows:

> [E]vents ... are classified into M event contexts. Each event belongs to one and only one event context. The number of event contexts, e.g., the number of packets received and the operations that are required on the packets is generally much larger than the number of execution contexts available (M >> N). The network processor 26 uses any available execution contexts to process unrelated events (those from different event contexts) in parallel while maintaining order in the processing of events within each of the event contexts.[2]
>
> ...
>
> Events arrive into the system via a global event queue 32 and are stored in per-execution context event queues 34. In the example discussed events are packets. A FIFO event

[1] The examples provided in this paper are for illustration only and should not be used to limit the scope of the claim.
[2] Page 4, lines 11-16 of Applicant's specification.

> queue 36 is associated with the execution context to temporarily store the events (packets) for that event context (for the duration of the binding). The events (packets) that are received by the network processor are dynamically bound on a per-event basis in the context queues 36.[3]

As described above, in an example implementation, an event context can be dynamically bound to an execution context to process events within that event context for the duration of the binding. An example of an execution context is a process or thread for performing a sequence of events.[4] An example advantage of dynamically binding an event context to an execution context is described in the Applicant's specification as follows:

> The network processor 26 uses any available execution contexts to process unrelated events (those from different event contexts) in parallel while maintaining order in the processing of events within each of the event contexts.

In Chen, a global output queue is shared by port output queues related to each of the ports for the purpose of merely forwarding packets in sequence to one or more ports.[5] In a bid to avoid sequence/data errors, Chen describes using a first-in first-out (FIFO) method to ensure that the packets are forwarded to the ports in sequence.[6] For example, Chen teaches allocating FIFO blocks to all of the port output queues and the global output queue. Based on a type of a received packet (i.e., unicast, multicast, or broadcast), the packet is forwarded to the ports using a predetermined mechanism of setting fields in the FIFO blocks corresponding to the queues.[7] As such, Chen fails to disclose or suggest "dynamically binding an event context to an execution context in response to receiving events."

The Examiner appears to rely on at least 4 columns in Chen as anticipating the features of Applicant's claim 1, without suggesting correspondences between the elements of claim 1 and the teachings of Chen. Should this rejection be maintained, the Applicant respectfully requests the Examiner to specifically identify exactly where Chen describes or suggests "dynamically binding an event context to an execution context in response to receiving an event by ... a global event queue that is accessible by event contexts ... storing events from the global event queue in per-execution context event queues and associating an event queue with the execution context... ."

---

[3] Page 5, lines 7-15 of Applicant's specification.
[4] Page 4, lines 1-4 of Applicant's specification.
[5] Column 5, lines 3-15 of Chen.
[6] Column 2, lines 21-32 of Chen.
[7] Column 1, lines 31-41, and column 11, lines 31-67, to column 12, line 9 of Chen.

For at least a similar reason given for claim 1, claims 27 and 42 are patentable.

Allowable subject matter

Regarding claims 12-14, the Examiner stated that these claims are objected to for being dependent on a rejected base claim, but are allowable if rewritten in independent form. The Applicant acknowledges that these claims are patentable.

Conclusion

All of the dependent claims are patentable for at least the reasons for which the claims on which they depend are patentable.

Canceled claims, if any, have been canceled without prejudice or disclaimer.

Any circumstance in which the Applicant has (a) addressed certain comments of the Examiner does not mean that the Applicant concedes other comments of the Examiner, (b) made arguments for the patentability of some claims does not mean that there are not other good reasons for patentability of those claims and other claims, or (c) amended or canceled a claim does not mean that the Applicant concedes any of the Examiner's positions with respect to that claim or other claims.

No fee is believed due. Please apply any charges or credits to deposit account 06-1050 referencing Attorney Docket No. 10559-0875001 / P17394.

Respectfully submitted,

Date: 5/6/09

Tonya S. Drake
Reg. No. 57,861

Fish & Richardson P.C.
225 Franklin Street
Boston, MA 02110
Telephone: (617) 542-5070
Facsimile: (877) 769-7945